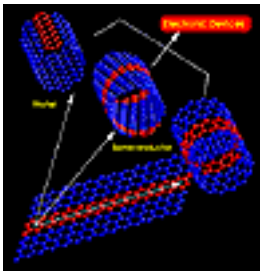
May-June 1997
Vol. 2, No. 24



Fullerene-based Nanoelectronic Devices Show `Tremendous Potential'

Researchers at the NAS Facility are examining the role of the fullerene structure -- a form of carbon -- in creating electronic devices for the next century. In this article, scientists Jie Han and Subhash Saini give a bird's-eye view of the transition from conventional CMOS-based semiconductor devices to quantum devices, and discuss how fullerenes may benefit `nanoelectronics' within the next ten years.

Need for Leading-edge Systems Spurs New Parallel Testbed

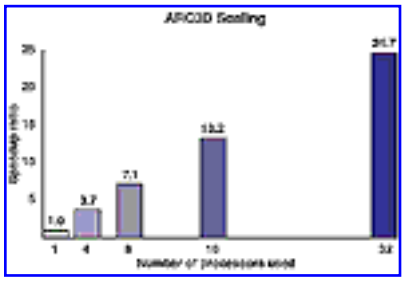
by Elisabeth Wechsler

The Ames Information Technology Program and NASA's Data Assimilation Office (DAO) have jointly purchased a Silicon Graphics Inc. (SGI) Origin2000 and plan to collaborate on systems software development and modification, as well as optimization of selected scientific codes for parallel computing. The project is the result of a memorandum of understanding between Ames Research Center and Goddard Space Flight Center to collaborate on information technology.

The $2.5 million hardware system with a theoretical peak performance of 25 gigaflops is one of SGI's new generation of high-performance, shared-memory multiprocessors (SMPs) based on the MIPS R10000 RISC microprocessor. While the system, which was installed in April at the NAS Facility, is officially designated as a testbed, it is expected to meet some aggressive performance targets once initial testing is completed later this month.

Performance of the GCM Code

System \ No. of CPUs	1 CPU	4 CPUs
SGI Origin2000	2104 seconds	602 seconds
CRAY J90	3208 seconds	752 seconds



"The NAS Parallel Benchmarks results from the Origin2000 system were excellent -- both in performance and price performance," said Bob Ciotti, of the high-speed processor group. "We also saw very good performance when we took a serial vector code (such as the global circulation model or ARC3D code) and parallelized and optimized it for this machine," he added.

Timing Is `Right'

With the delay of the NAS Facility's next generation high-speed processor and the recent shift in emphasis for the CRAY J90 (newton) project away from parallel processing, the timing couldn't be better for the partnership.

For DAO, there were several persuasive considerations that brought this project to reality, according to Jose Zero, DAO systems engineer and software architect. Based at Goddard, DAO's role within the NASA Mission to Planet Earth Project is to produce globally coherent (physically and chemically) datasets for the atmosphere, ocean, and land surfaces.

DAO has been running high-performance codes on the forerunner of the Origin2000, the SGI POWERCHALLENGE Array system. The organization's goal is to maximize performance on

commodity-based parallel systems that require less capital investment than traditional vector supercomputers.

"Our science is strictly limited by our computing capacity, so we want to achieve as many floating point operations per second as possible within our budget," Zero said.

Another factor, he explained, is the long lead time -- "many years" -- needed to develop useful software. DAO wants to maximize the life span of its software by adapting and porting it without substantial recoding to newer and faster hardware as soon as it becomes available. With this payoff in mind, Zero said that his office is willing to work with the anticipated instability of a new system.

Access to Cutting-edge Systems

This need for access to cutting-edge systems provided the logical key to collaborate with Ames, especially because of the NAS Facility's leadership in pathfinding research in supercomputing and user support. (Ames is designated as the Center of Excellence for Information Technologies for NASA.)

Zero sees DAO as a "special user" -- one with high-performance low-cost needs, as well as an interest in participating in the full cycle of hardware and software maturation. "We're not the sort of user who just gets an account and runs jobs," he noted. "We also want to participate with the vendor in the process of maturing the platform. That way, problems show up quickly and -- hopefully -- are resolved quickly."

The new system combines the best of both SMP and high-performance scalable architectures, according to Nateri Madavan, a scientist at the NAS Facility involved with the Origin2000 project. "One of the system's nice features is the distributed shared-memory architecture, in which main memory is physically placed among all the processors but accessible to and logically shared by all of them," he said. "This single-address space makes the system very easy to program by allowing the application to use all of the processors and main memory without explicit message passing."

"Users accustomed to multitasking their codes on the NAS Facility's CRAY C90 should feel very comfortable with the Origin2000," Madavan said. "Unlike the C90, however, memory access times for a processor will not be uniform and will depend on how physically remote the memory is from the given processor."

"To achieve the maximum potential of this machine, users will have to modify data structures and optimize their codes to account for the multilevel hierarchical memories of RISC microprocessors. Luckily, many of the optimization techniques used on vector processors also work quite well on cache-based RISC processors," he added.

Easier To Program

In general, the other two parallel testbeds installed at the NAS Facility "haven't gained wide acceptance among users," Madavan commented. A 160-node IBM SP2 (babbage, installed almost three years ago) is a distributed-memory system that has been "difficult to program because it requires message passing to communicate between nodes. And a 36-node SGI POWERCHALLENGE Array, (davinci, almost two years old) is a cluster of bus-based SMPs that suffers serious performance degradation because of the limited bandwidth of the bus," he continued.

"By moving away from a bus-based approach, the Origin2000 potentially removes the main bottleneck to performance and scalability of the POWERCHALLENGE Array," Madavan explained. "This, together with its easy-to-use shared-memory programming model, makes the system look very promising."

The 64-node Origin2000 system -- the first of its kind to be run as a single system, according to Mary Hultquist, Origin2000 project lead -- has been named "Alan Turing," after the well-known mathematician and computer scientist. The system architecture is quite flexible, allowing for systems as small as 16 processors to be operated independently or as a single 64-node system. Another 64 nodes (for a total of 128 nodes) will possibly be added later, Hultquist said.

Another possibility is for DAO and the NAS Facility to create a metacenter of Origin2000 systems at other NASA centers. This would provide a single entry point to access all the systems, rather than load-balancing across several systems (the concept used for the Ames Metacenter project, a joint effort between the NAS Facility and Langley Research Center.

In the meantime, the NAS Facility will manage the testbed portion of the collaboration, with up to 20 percent of the dedicated runtime devoted at least initially to testing and development, Hultquist said, adding that this figure "will be adjusted as the system becomes more stable."

User Requirements

Users will be selected according to their willingness to adapt codes to the testbed system and work with the NAS Facility staff to understand the strengths and weaknesses of the architecture.

"We're looking for users who will be a good match for this project -- those with applications that are important to NASA who are willing to parallelize and/or optimize their codes," Ciotti said.

"Our goals are similar to DAO's and we're excited about their willingness to share the risk of investigating this new technology," he added.

If the Origin2000 performs well, the system may be used to increase capacity to the Information Technology Program for FY98, Ciotti said. "We hope to have a good assessment of this architecture by October."

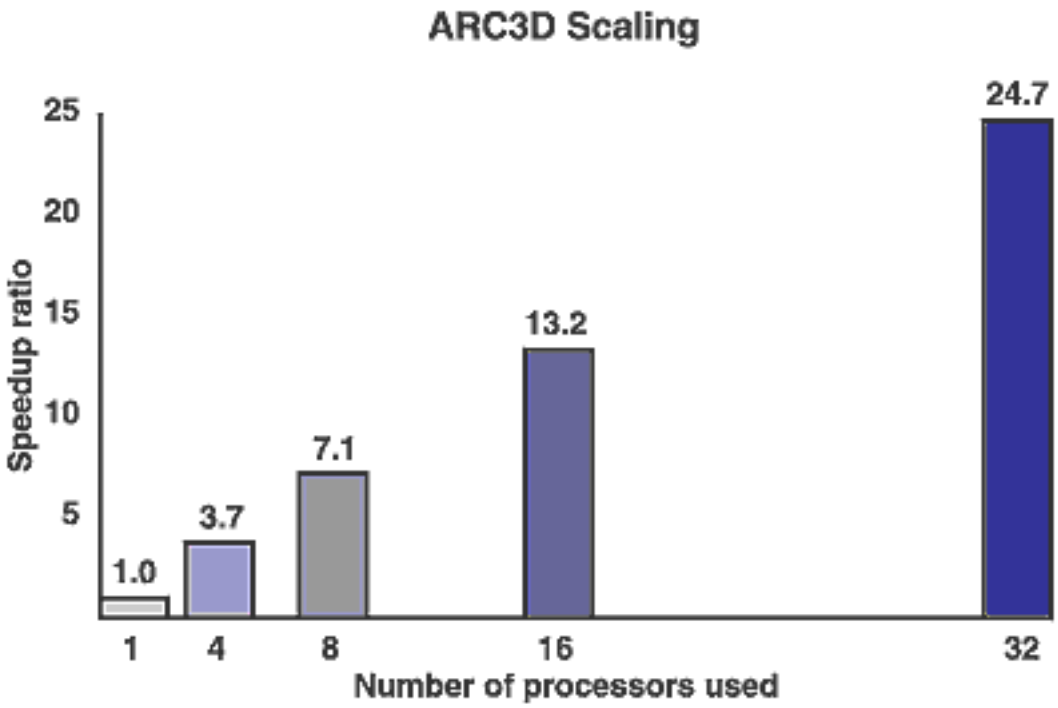
To find out more about participating in the Origin2000 testbed project, contact Mary Hultquist.

Performance of the GCM Code

System \ No. of CPUs	1 CPU	4 CPUs
SGI Origin2000	2104 seconds	802 seconds
CRAY J90	2208 seconds	752 seconds

This table shows the performance of the global circulation model (GCM) code. With little optimization, the Origin2000 version of GCM matches the CRAY J90 system at the NAS Facility in performance. The J90 version has been extensively optimized, and the Origin2000 system costs substantially less per processor than the J90 system.

 to the article



Benchmarks based on runs made in March 1997 on an SGI Origin 2000. These tests showed that the ARC3D code, when extensively optimized, shows an excellent speedup ratio (scaling) through 32 processors.

to the article

NAS scientist David Bailey gives us a glimpse at the discovery of several new math formulas that managed to elude mathematicians until recently. Bailey and others have used `experimental mathematics,' a method that takes advantage of supercomputing technology to find and prove these new formulas. For those who don't remember the math -- we know you're out there -- just skip it and read the rest of this intriguing article.

New Math Formulas Discovered With Supercomputers

by David H. Bailey

In April 1993, Enrico Au-Yeung, an undergraduate at the University of Waterloo, brought to the attention of Jonathan Borwein, his professor, the curious fact that:

$$\sum_{k=1}^{\infty} \left(1 + \frac{1}{2} + \cdots + \frac{1}{k}\right)^2 k^{-2} \approx 4.59987 \approx \frac{17\pi^4}{360}$$

based on a computation to 500,000 terms. Borwein was skeptical of this finding -- if there was such an identity, why hadn't the theory behind it been discovered by mathematicians centuries ago? Borwein tried computing this sum to a higher level of precision in order to demonstrate to the student that this conjecture really did not precisely hold.

Surprisingly, in subsequent computations by Borwein to 30 digits and by myself to over 100 decimal digits, this relation *was* upheld. Needless to say, it is rather unlikely that a mathematical relation could hold to such high precision and yet not really be true.

Intrigued by this finding, Borwein, with myself and other researchers, subsequently discovered numerous other identities of this form. Many, but not all of these specific results have subsequently been proven rigorously.

Experimental Mathematics

Our methodology, termed "experimental" mathematics, is as follows:

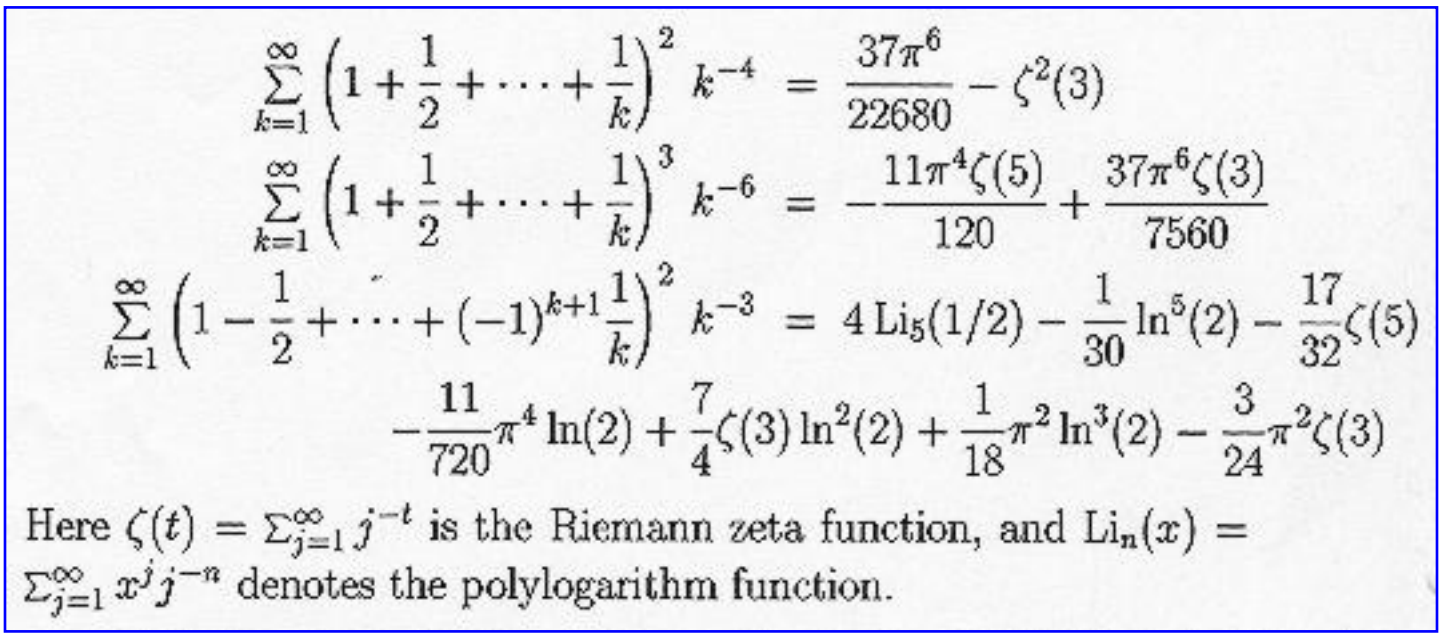
1. Use computer programs to perform searches, finding tentative results.
2. Study these results for patterns to suggest further searches.
3. Formulate rigorous proofs for these specific results.

4. Extend specific proofs to general mathematical results.

The computer searches required the values of these sums and constants to be computed to very high precision -- from 100 to 1,000 decimal digits. Computing these values to such high precision required advanced numerical techniques, and was usually quite expensive for these true supercomputer class problems. Once these values were computed, we then applied an "integer relation" algorithm. This is an algorithm that, when given an *n*-long vector of real numbers *x*sub_i, attempts to find integer coefficients *a*sub_i so that:

$$a_1x_1 + a_2x_2 + a_3x_3 + \cdots + a_nx_n = 0$$

I have used the "PSLQ" integer relation algorithm, which was discovered by Helaman Ferguson, of the Center for Computing Sciences in Maryland.



$$\sum_{k=1}^{\infty}\left(1+\frac{1}{2}+\cdots+\frac{1}{k}\right)^2 k^{-4} = \frac{37\pi^6}{22680} - \zeta^2(3)$$

$$\sum_{k=1}^{\infty}\left(1+\frac{1}{2}+\cdots+\frac{1}{k}\right)^3 k^{-6} = -\frac{11\pi^4\zeta(5)}{120} + \frac{37\pi^6\zeta(3)}{7560}$$

$$\sum_{k=1}^{\infty}\left(1-\frac{1}{2}+\cdots+(-1)^{k+1}\frac{1}{k}\right)^2 k^{-3} = 4\,\mathrm{Li}_5(1/2) - \frac{1}{30}\ln^5(2) - \frac{17}{32}\zeta(5)$$
$$-\frac{11}{720}\pi^4\ln(2) + \frac{7}{4}\zeta(3)\ln^2(2) + \frac{1}{18}\pi^2\ln^3(2) - \frac{3}{24}\pi^2\zeta(3)$$

Here $\zeta(t) = \Sigma_{j=1}^{\infty} j^{-t}$ is the Riemann zeta function, and $\mathrm{Li}_n(x) = \Sigma_{j=1}^{\infty} x^j j^{-n}$ denotes the polylogarithm function.

Some identities that we have found using this approach are shown in the identity. While these formulas were originally studied for purely mathematical interest, other researchers have subsequently found applications for them in the field of quantum physics, and research into additional formulas continues.

Historical Fascination With pi

Humankind has been fascinated with the constant pi = 3.14159... throughout recorded history. The Babylonians used the approximation 3.125, while the Old Testament (I Kings 7:23). gives the value 3.0. Archimedes found the first true algorithm for computing pi in about 250 BC, and used it to find pi to about four digits accuracy. Isaac Newton found several new formulas for pi. Using these formulas, he recorded a value of pi to 16 decimal places in his notebook, but later sheepishly admitted, "I am ashamed to tell you how many figures I carried these computations, having no other business at the time."

More recently, modern computer technology has made it possible to compute millions and even billions

of digits, by employing fast new formulas for pi and efficient arithmetic techniques, based on the fast Fourier transform (FFT). In the most recent computation of this sort, Yasumasa Kanada at the University of Tokyo computed over 6.4 billion decimal digits of pi.

Although there are no "practical" motivations for these computations, there is some mathematical interest. In particular, although mathematicians suspect that the digits of pi (as well as numerous other math constants) are "random," they have never been able to prove this assertion in even a single instance.

Thus there is continuing interest in the output of such calculations, to see if there are any statistical irregularities that would suggest that this conjecture is false. And there have been some practical benefits of large pi calculations -- for example, some advances in efficient methods for computing FFTs had their roots in attempts to perform extra-high-precision arithmetic.

A New Formula for pi

Through centuries of such investigations and calculations, mathematicians have assumed that the computational cost of computing just the *n*-th digit of pi and similar constants is not significantly less than computing all of the first n digits. In other words, it has been assumed that there is no "shortcut" to computing just the *n*-th digit of pi.

Thus, it came as no small surprise when two Canadian mathematicians and I recently found exactly such an algorithm. In particular, this simple scheme allows one to compute the *n*-th hexadecimal digit of pi without computing any of the first *n*-1 digits, without using multiple-precision arithmetic software, and with very little memory. The 1,000,000-th hex digit of pi can be computed in just two minutes on a personal computer.

This scheme is based on the following remarkable new formula for pi:

$$\pi = \sum_{k=0}^{\infty} \frac{1}{16^k} \left(\frac{4}{8k+1} - \frac{2}{8k+4} - \frac{1}{8k+5} - \frac{1}{8k+6} \right)$$

Like the formulas shown above, this one was discovered using the PSLQ integer relation algorithm. This is probably the first instance in history that a significant new formula for pi was discovered by a computer. Furthermore, the proof is so simple that anyone who has completed a good first-year calculus course should be able to follow it. One wonders why this formula (and proof) wasn't found, say, by Euler over 200 years ago.

Computing the *n*-th hexadecimal digit of pi using this formula is also easy. Mathematicians have not found any formula of this form for computing the *n*-th *decimal* digit of pi, and many mathematicians believe that there is no base-ten formula.

Computers Will Expand Vision

What will the future bring? No one believes that formal, rigorous mathematical proofs will become obsolete any time soon. But it does seem clear that computer technology will be used much more by the next generation of pure mathematicians, greatly expanding their vision.

For more information on these new math formulas, as well as simplified C and Fortran programs for performing some of these computations, send email to **dbailey@nas.nasa.gov**.

$$\sum_{k=1}^{\infty} \left(1 + \frac{1}{2} + \cdots + \frac{1}{k}\right)^2 k^{-4} = \frac{37\pi^6}{22680} - \zeta^2(3)$$

$$\sum_{k=1}^{\infty} \left(1 + \frac{1}{2} + \cdots + \frac{1}{k}\right)^3 k^{-6} = -\frac{11\pi^4\zeta(5)}{120} + \frac{37\pi^6\zeta(3)}{7560}$$

$$\sum_{k=1}^{\infty} \left(1 - \frac{1}{2} + \cdots + (-1)^{k+1}\frac{1}{k}\right)^3 k^{-3} = 4\,\mathrm{Li}_5(1/2) - \frac{1}{30}\ln^5(2) - \frac{17}{32}\zeta(5) - \frac{11}{720}\pi^4\ln(2) + \frac{7}{4}\zeta(3)\ln^2(2) + \frac{1}{18}\pi^2\ln^3(2) - \frac{3}{24}\pi^2\zeta(3)$$

Here $\zeta(t) = \sum_{j=1}^{\infty} j^{-t}$ is the Riemann zeta function, and $\mathrm{Li}_n(x) = \sum_{j=1}^{\infty} x^j j^{-n}$ denotes the polylogarithm function.

$$\sum_{k=1}^{\infty} \left(1 + \frac{1}{2} + \cdots + \frac{1}{k}\right)^2 k^{-4} = \frac{37\pi^6}{22680} - \zeta^2(3)$$

$$\sum_{k=1}^{\infty} \left(1 + \frac{1}{2} + \cdots + \frac{1}{k}\right)^3 k^{-6} = -\frac{11\pi^4\zeta(5)}{120} + \frac{37\pi^6\zeta(3)}{7560}$$

$$\sum_{k=1}^{\infty} \left(1 - \frac{1}{2} + \cdots + (-1)^{k+1}\frac{1}{k}\right)^2 k^{-3} = 4\,\mathrm{Li}_5(1/2) - \frac{1}{30}\ln^5(2) - \frac{17}{32}\zeta(5)$$
$$-\frac{11}{720}\pi^4\ln(2) + \frac{7}{4}\zeta(3)\ln^2(2) + \frac{1}{18}\pi^2\ln^3(2) - \frac{3}{24}\pi^2\zeta(3)$$

Here $\zeta(t) = \Sigma_{j=1}^{\infty} j^{-t}$ is the Riemann zeta function, and $\mathrm{Li}_n(x) = \Sigma_{j=1}^{\infty} x^j j^{-n}$ denotes the polylogarithm function.

How to Compute an Arbitrary Hex Digit of pi

Let *S*sub_1 be the first term in the new formula for pi. Note that:

$$\pi = \sum_{k=0}^{\infty} \frac{1}{16^k}\left(\frac{4}{8k+1} - \frac{2}{8k+4} - \frac{1}{8k+5} - \frac{1}{8k+6}\right)$$

$$S_1 = \sum_{k=0}^{\infty} \frac{1}{16^k(8k+1)}$$

$$\mathrm{frac}(16^d S_1) = \sum_{k=0}^{\infty} \frac{16^{d-k}}{8k+1} \pmod 1$$

$$= \sum_{k=0}^{d} \frac{16^{d-k} \pmod{8k+1}}{8k+1} + \sum_{k=d+1}^{\infty} \frac{16^{d-k}}{8k+1} \pmod 1$$

1. For each term in the first summation of the last line, perform the exponentiation in the numerator using the binary algorithm for exponentiation, reducing mod 8*k*+1 after each multiplication operation.

   Divide the resulting numerator by the denominator 8*k*+1, and then add the result to the sum, discarding the integer part after each addition.

2. Evaluate the second summation in the last line. Only a few terms need to be evaluated before convergence to 64-bit IEEE accuracy.

   Add to the result of the first summation and discard integer part.

3. Repeat steps 1 and 2 for each of the four sums in the formula for pi.

   Calculate 4*S*sub_1-2*S*sub_2-*S*sub_3-*S*sub_4 as indicated in the formula for pi.

   Discard the integer part, and if negative, add 1.

4. The resulting fraction, when converted to hexadecimal, gives the hex digits of pi beginning at position *d*+1 up to about position *d*+8. Past position *d*+8, the digits may not be accurate due to round-off errors.

The above algorithm can be performed exclusively using ordinary 64-bit IEEE arithmetic, provided that $d < 2^{26}$. As an example, the hex digits of pi from positions 1,000,000 through 1,000,007 are 26C65E52.

 to the article.

Using the /big Scratch Directory on newton

By George B. Myers

The recently reconfigured newton system (four CRAY J90s) is now operational. This article focuses on ways to ensure that users get the best job turnaround -- primarily through the use of the scratch filesystem known as /big. Several other areas that may be confusing to new users will also be touched on.

Maximum File Space Provided

The newton home filesystem is patterned after the "superhome" filesystems used on the NAS and Aeronautics Consolidated Supercomputing Facility (ACSF) CRAY C90s, vonneumann and eagle. (See "Filesystem Reconfiguration Makes Sense for ACSF Users," NAS News, March-April.) This filesystem provides maximum file space for users by utilizing Cray's Data Migration Facility to migrate large files -- those over 1 megabyte -- to quickly accessible silo tape storage. This setup allows each user to have 10 thousand files and 1 gigabyte (GB) of local storage space.

On the newton system, the home filesystem is cross-mounted between the four J90s using the Network Filesystem (NFS). Interactive use of newton is restricted to one J90 (newton1), where the home filesystem resides locally. This way, file access for interactive processing is not hindered by the inherent slowness of NFS. The high speed processor group considers this the best way to provide a common home filesystem for all processing nodes in the newton system.

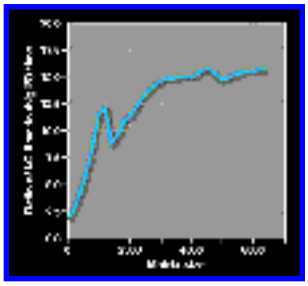
The one drawback to this configuration is that access to files via NFS is very slow. To compensate, the managed scratch filesystem known as /big is included in the configuration. This Session Reservable Filesystem (SRFS) allows users to reserve file space for running a job, guaranteeing that this scratch space will be available at the time the job runs. In addition, the /big filesystem resides on each J90 system, which means that file I/O will be faster than to the home filesystem via NFS.

One problem with other methods of providing local scratch space is that there's no way to guarantee space availability when the job runs. On the other hand, /big is a temporary filesystem, which means that you must copy those files you want to keep to another place for storage. Your home file space, if managed properly, should easily provide this space. In addition, every user has access to longer-term storage space on the NAStore system, chuck.

Using /big is Essential

Johnny Chang, a consultant in the NAS Systems Division's scientific consulting group, ran some simple

experiments exercising I/O on each J90. The test results indicate that I/O to the home filesystem is anywhere from 2-16 times slower than to the /big filesystem from newton2 to newton4. The same tests on newton1 (where the home filesystem resides) gave pretty much the same results for /big and home -- not surprising, since the home filesystem is NFS-mounted on the other three J90s. This points out how important it is to make use of the /big filesystem.



The test data also indicates that as the amount of I/O goes up, so does the system time spent processing the I/O. This costs users more time and slows the whole system down. These tests were simple reads and writes of entire matrices of ever-increasing size. (See graph.)

The conclusion: use the /big filesystem for your batch job I/O to ensure the minimum and most consistent time to process jobs.

How to Use the /big Filesystem

An environment variable called $BIGDIR is created automatically at interactive login time and at the beginning of batch jobs. This variable points to a directory, which is accessible by only that instance of login or that batch job, in the /big filesystem. Use $BIGDIR to then copy files, reference files, and change to the directory in the /big filesystem residing on the system where your jobs will run.

The following PBS (Portable Batch System) script illustrates how to request or reserve file space on the /big filesystem. Remember that required files are copied to the /big filesystem for use during job processing and are then copied back to the home filesystem at the end of the job for safekeeping. Changing directories to the /big filesystem ensures that any files created during the job will be created in /big.

```
#PBS -S /bin/csh
#PBS -l cput=4:00:00
#PBS -l pcput=3:58:00
#PBS -l mem=200MW
#PBS -l srfs_big=250MW
#PBS -q mt

# copy input files to $BIGDIR
rcp newton1:bigcase/datain/basegrid $BIGDIR
rcp newton1:bigcase/datain/caseinput $BIGDIR
# etc.
rcp newton1:code/execs/flowsolver $BIGDIR

cd $BIGDIR
```

```
# execute code
ja
./flowsolver < caseinput > case.out
ja -st

# copy output files to be saved
rcp case.out newton1:bigcase/dataout/
rcp restart newton1:bigcase/dataout/
rcp lastgrid newton1:bigcase/dataout/
# etc.
```

Script Explanation

Specifying a process time limit (*pcput*, line 3 in the script) that is slightly less than the job time limit (*cput*, line 2) allows you to do some processing in the event that the job runs out of time while executing the main process. For instance, this time can be used to save the last restart file.

The commands *ja* and *ja -st* provide useful job accounting information, including command names and their corresponding elapsed time, user CPU time, and system CPU time. (See the man pages for both commands for details.)

Use the remote copy command (*rcp*) simply to avoid using NFS.

Output files are copied back to the home filesystem for temporary storage until they can be moved to NAStore or elsewhere for long-term storage.

For more information on this and other topics, contact User Services at (415) 604-4444 or (1-800) 331-8737, or send email to **nashelp@nas.nasa.gov**.

Quick Tips For New HSP Users

For those who are new to the NAS Facility and ACSF systems, here are a few extra tips for working in these environments.

A Word on PBS

PBS, the Portable Batch System is now run on all the NAS Facility high-speed processors and testbeds. (Previously, NQS, Cray's batch scheduler, was used). PBS commands are very similar to NQS, however, the resource flags and parameters are different, as shown in the script. (See *NAS News*, March-April '96.)

The Module System

Cray Research has implemented a system for controlling licensed products that makes it fairly simple to distinguish between multiple versions of their products. For example, a reported problem may be fixed in a newer version, but that version has not been made the default version.

The module command provides a way to "switch" to the newer version to see if a problem that you're having has been fixed. More extensive information on this topic is available).

This information applies to vonneumann and eagle, as well. If you're having problems accessing the compilers, be sure to check this online document and review the instructions for setting up your environment in the section "All About Newton."

Reminder: Convert to Fortran 90

As stated in previous issues of *NAS News*, Cray Research no longer supports Fortran 77 -- although the NAS Facility still has the cf77 compiler. We strongly suggest that you begin converting your code to Fortran 90. An online tutorial on Fortran 90 is available.

 to the article

Filesystem Reconfiguration Makes Sense for ACSF Users

by Jill Dunbar

In a move to give users of the Aeronautics Consolidated Supercomputing Facility (ACSF) more flexibility in managing their home filesystems, the NAS Facility's high speed processor (HSP) group is reconfiguring the current filesystem. The change, a process which is expected to continue through June, will also make the system easier to use, noted Daniel DePauk, HSP systems analyst.

DePauk explained that users are sometimes confused with the current two-filesystem setup on the ACSF CRAY C90 (eagle), consisting of home directories for storing small files and source code, and a directory in the Central Storage Facility (CSF) for housing large datasets for the long term.

The new filesystem structure will mirror that already in place on vonneumann, the NAS Facility's CRAY C90: a "superhome" hybrid filesystem that allows small files -- as well as large files needed in the short term -- to remain accessible, or "online." Files headed for long-term storage will be stored in the NAStore mass storage system.

The decision to convert to a superhome system ties in with upgrades to disk storage hardware, which should take place within about six months, DePauk said.

Users Need To Know:

He emphasized that users need to be aware of three items:

- All home filesystems have already been configured as the superhome filesystem.
- Users are encouraged -- and have been since last fall, DePauk noted -- to move their current working files that now reside on the CSF, to their home directories on eagle.
- Rarely accessed files should be copied from the CSF to NAStore for long-term storage; then, these files and unneeded files should be removed from eagle.

"It's preferable for users to move their own files from CSF as soon as possible," DePauk said. Otherwise, the HSP group will move these files to users' home directories on scott (one of the NAStore machines) in a directory named eagle_csf. "After that, the files will no longer be accessible by users on eagle."

Alan Powers, HSP group lead, estimated that about 1.5 million files will be moved. Of these, about 90 percent -- all files of less than 1 megabyte (MB) -- will always stay online. He noted that that amount

represents less than 5 percent of the data stored.

HSP group members will notify users by email or phone one or two days in advance of file moves. Those who can't remember their scott passwords can retrieve them by contacting ACSF/NAS User Services: (415) 604-4444 or (800) 331-8737, email **nashelp@nas.nasa.gov**.

Definitions, Limitations

Elaborating on the use of superhome and NAStore, DePauk stated the following guidelines:

- "Small" files are defined as those under 1 megabyte (MB). As file space is needed, those larger than 1 MB may be migrated from disk to tape by DMF, Cray's Data Migration Facility. The presence of DMF guarantees that any file under the 1-MB limit (such as source code) will always be online.

- "Short-term" storage (for larger files used by current projects) is loosely defined as a maximum of one month.

- Unused files that need to be retained for long periods should be moved to NAStore and deleted from eagle.

In addition, each user receives a generous disk quota of 10 thousand files and 1 gigabyte of file space.

Batch Jobs Get 'Special' Treatment

For batch jobs, users are cautioned to not use their home directories as the current working directory because larger files have poor I/O performance in home directories, DePauk explained. Users will get better job turnaround if they use the following alternate procedure:

1. Copy the file to $BIGDIR (as the current working directory.
2. Change directories to $BIGDIR.
3. Execute the program.
4. After the jobs complete, copy needed files back to the home directory.

See information on the progress of the ACSF filesystem migration. or send email to hsp-snp@nas.nasa.gov.

AFLIC, the Animated Flow Line Integral Convolution software, produces surface flow patterns that resemble results from wind tunnel experiments (see NAS News, November-December '96). In this followup article, AFLIC co-developer David Kao reports on the beta version and discusses a new program that he and colleague Han-Wei Shen are developing for visualization of unsteady surface flow.

CFD Users Give Positive Feedback On AFLIC Software

by David Kao

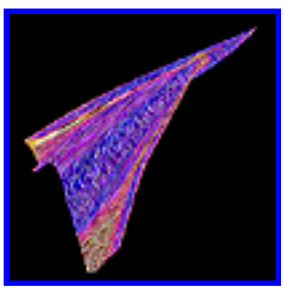
Since its beta release last December, about a dozen CFD researchers have used AFLIC (Animated Flow Line Integral Convolution) software to visualize surface flows from their numerical simulations. Preliminary feedback from these researchers has been very positive.

One reason users like AFLIC is that it generates continuous flow lines, which produces smoother surface flow on the model body. In contrast, traditional methods use particle traces and vector plots to visualize surface flows. The flow patterns produced by these other techniques are discrete, and artifacts may be introduced due to grid density.

User Testing, Feedback Valuable

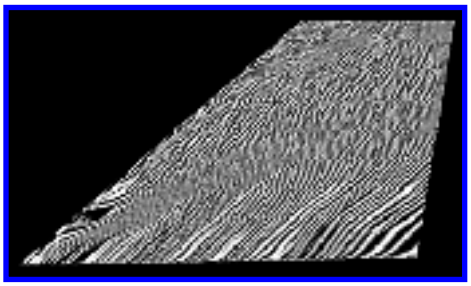
Users also like AFLIC's ability to produce surface flows that resemble those in flow experiment visualization (wind tunnel tests). Stuart Rogers, a researcher in the Design Cycle Technologies Branch at Ames Research Center, has used AFLIC to generate a variety of surface flows on a high-powered lift aircraft. Rogers is working on a project that uses computational fluid dynamics to compute the flow over an entire aircraft in a high-lift configuration.

"These geometries are extremely complex, as are the fluid dynamics in these flowfields," Rogers explained. "These configurations typically require grid systems with 15-30 million points, and postprocessing of such flowfield solutions is very difficult. The pictures generated by AFLIC can be helpful in determining the flowfield topology."



Sundaram Pichuraman, at McDonnell Douglas Aerospace, has used AFLIC for quick flow analysis of several problems in propulsion/airframe integration and other related CFD work. Specifically, these cases have significant flow separation and reattachment regions (see figure at left). "The [AFLIC] code can be run in conjunction with an analysis code, providing important flow details in different regions of the flow field -- while the solution is in progress -- to provide diagnostics

of the flow development," Pichuraman said.



Another example of ALFIC's use is illustrated in the figure at right, which shows surface flow on the tail of an F-18 aircraft.

These users have offered suggestions for improving AFLIC's value to researchers; for example, making additional scalar functions available for shading flow textures, and adding a GUI (graphical user interface) panel for interactive control of user parameters. Some of these suggestions will be implemented in the next version of AFLIC, planned for release at the end of the year.

Unsteady Flow Program Needed

At present, AFLIC provides instantaneous visualization of steady surface flow; that is, the program computes the surface flow based on the velocity field at an instant in time. Although AFLIC can be used for unsteady flows by first computing the surface flow at every time step -- one step at a time -- and then animating the computed surface flows from each time step, this method may not capture the time-varying phenomena accurately.

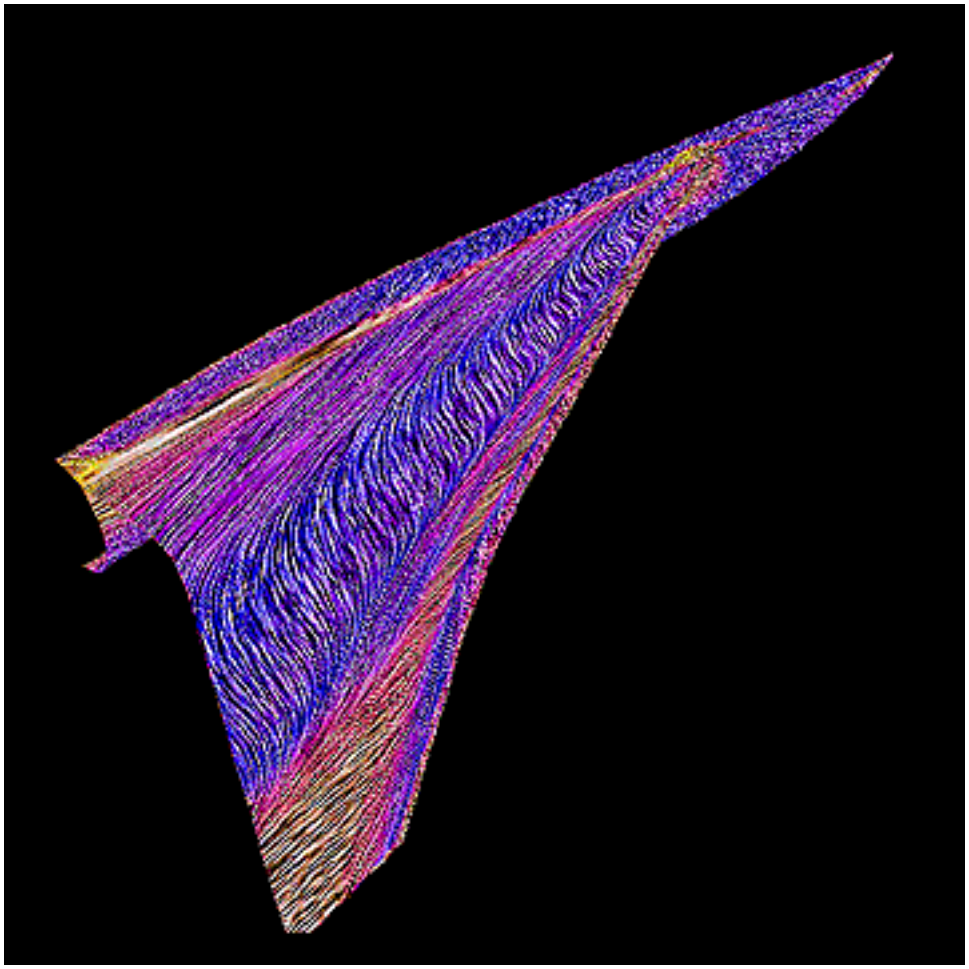
For example, when the velocity field has spiraling vortices, animating the surface flows with AFLIC will only show translation (that is, the displacement of the vortices) but not the spiraling motion of the vortices. Another example is that animating instantaneous surface flows may not accurately reveal the interaction of the flows over time, such as vortex formation and shedding. These types of inaccuracies emphasize the need for visualizing unsteady surface flows.

New Algorithm for Unsteady Flows

Han-Wei Shen, who recently joined the NAS Systems Division's data analysis group, has been working with me to develop a new technique for numerical surface flows for unsteady flows. The result is a new program, called UFLIC, which accurately presents the time-varying phenomenon of unsteady flows.

Although UFLIC is still under development, some promising results have been obtained. UFLIC employs a new algorithm that takes into consideration the time variable and flow patterns from previous time steps. In unsteady flows there are no well-defined flow lines because the flow is dynamic and particles can advance to new locations easily. The new algorithm uses a time-dependent particle tracing scheme to trace the flow textures, as well as image processing techniques to improve image quality.

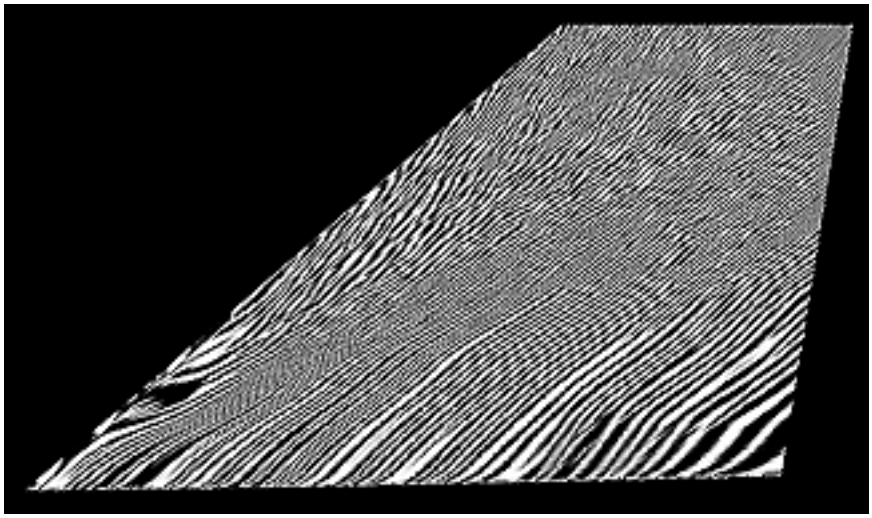
The first version of UFLIC is planned for release by the end of the year. To become an AFLIC user, send email to **davidkao@nas.nasa.gov**.



This image, which was generated using AFLIC (the Animated Flow Line Integral Convolution software), reveals the surface flow pattern of a wing/body configuration at transonic speeds for a high negative angle-of-attack.

Image courtesy of Sundaram Pichuraman, McDonnell Douglas Aerospace.

to the article.



Close-up of survace flow over the vertical tail of a twin-tailed F-18 aricraft at high angle-of-attack. The image, generated by AFLIC software, reveals unsteady flow behaviour due to bursting of the leading edge extension vortex along the aircraft's body.

Dataset by Ken Gee, NASA Ames Design Cycle Technologies Branch.

to the article.

Collaboration Will Create Standards for Parallel Job and Resource Scheduling

by Parkson Wong

As part of a Cooperative Research Agreement, the NAS Facility -- along with several NASA centers, IBM, Pratt & Whitney, and Platform Computing, among others -- has begun developing a standard interface for parallel job scheduling and task management. Currently, most job management systems (such as LSF, LoadLeveler, and Condor) tightly couple different modules into a single integrated product.

These modules handle starting new tasks, scheduling running jobs, managing resources, and interacting with the message-passing library. By providing standard interfaces between these complex components, this standards effort, known as PSCHED, will provide the framework for the following:

- vendor interoperability -- the message passing interface (MPI) library from one vendor will work with the batch queuing system from another vendor.
- architecture independence -- a single job scheduler can be written, which will manage shared-memory multiprocessor (SMP), Parallel Virtual Machine (PVM), and MPI jobs on a variety of architectures, regardless of the batch queuing system installed.
- improved interaction with the research community -- a research scheduler can be directly plugged into a production system for testing and verification.

The project has been driven by the relatively new metacenter concept, which pools a variety of machines located at different facilities to schedule and run users' jobs. This concept, which the NAS Facility has been closely involved with since spring 1995, has encountered several technical challenges, including:

- Some schedulers are tightly integrated with a specific message-passing library, such as Condor and PVM.
- Almost all schedulers are tightly integrated with specific batch queuing systems.
- A lack of support for parallel jobs.

The goal of the PSCHED applications programming interface (or API) is to allow any site to write a scheduler that can schedule a variety of parallel jobs to run on a collection of different machines. To achieve this goal, standardization of the interfaces between the different modules -- message-passing libraries, task manager, resource manager, and scheduler -- is necessary (see figure, right).

Specific Module Roles

The specific roles of these modules are:

- task manager -- Provides task management services, such as: spawn a task on a node, either local or remote; deliver a signal from one task to another task within the same parallel application; and interface with a resource management function to provide information about nodes, to obtain additional resources (nodes), to free nodes no longer required, and to notify tasks of the need to checkpoint, suspend, and/or migrate.

- resource manager -- Provides resource management services such as: monitor the resources available in the system, reserve or allocate resources for tasks, and release or de-allocate resources no longer needed by tasks.

- scheduler -- Schedules jobs; responsible for determining which task should be run on the system according to site-specific policies and the resources available in the system.

Minimal Functionality Will Be Identified

PSCHED is not an effort to standardize *how* any of these modules should be implemented. Rather, it is an effort to identify the minimal functionality needed from each module and then standardize its interface. For example: the scheduler is a user of the interfaces provided by the task manager and the resource manager; the scheduler waits for scheduling events from the task manager.

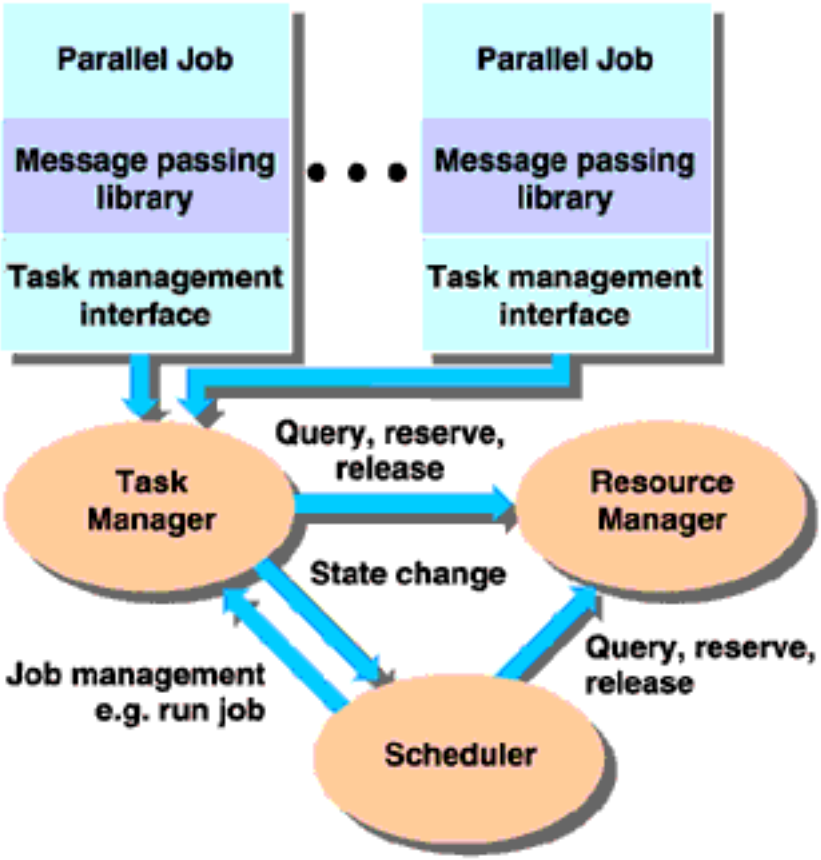
PSCHED is divided into two areas: The first is a set of calls for use by parallel processing jobs to spawn, control, monitor, and signal tasks under the control or management of the job/resource management system. This set of calls should meet the needs of MPI-2, PVM, and other message-passing implementations.

The second area is set of calls to be used by batch job schedulers. These calls will allow the development of a consistent job/resource scheduler -- independent of the job/resource management system used. The calls are intended to provide a standard way of obtaining information about the resources available in the processing environment and about the supply of jobs and requirements.

Progress To Date

The PSCHED effort began last August after a kick-off meeting held during the Computational Aerosciences Workshop held at Ames Research Center. At the Supercomputing '96 conference in Pittsburgh, a draft task management interface was released in a birds-of-a-feather session. Recently, a draft of the scheduler-related interfaces was released on the World Wide Web, where an ongoing email discussion, the draft documents, and other information about PSCHED are located.

For more information on PSCHED, send email to **parkson@nas.nasa.gov**.



The components and interfaces of parallel resources scheduling systems.

Artwork by Gail Felchle.

to the article.

Recent Technical Seminar Videos Are Available

Most research talks given at the NAS Facility are available through the NAS Documentation Center videotape loan program, including training events. Procedures for obtaining training event materials are available online, or send email to **doc-center@nas.nasa.gov**.

Here are summaries of the most widely attended technical seminars at the NAS Facility for the first quarter of 1977.

Nanotube Devices: Modulation of Electronic and Structural Properties of Carbon Nanotubes through Dimensionality, Hybridization, and Boundary Conditions, presented by Vincent Crespi, of UC Berkeley's Physics Department. Carbon nanotubes can be semiconductors or metals depending on the boundary conditions around the tube. Defects within the tube can either induce transitions between metal and semiconductor, or change the semiconducting band gap. Nanoscale devices formed from carbon nanotubes are proposed, and the synthesis and structural properties of carbon nanotubes are discussed. (3/6/97)

Computer-aided Software Evolution, presented by Valdis Berzins, of the Naval Postgraduate School. Concepts and tools for achieving software flexibility developed for computer-aided prototyping and software architectures are described. Automation support for flexibility of complex software systems depends on the integration of many factors, including simple and expressive modeling primitives, natural notations for describing desired system functionality, generic software architectures, and automatic program generation relative to those notations -- to name a few.

Improvements in technology for engineering databases have been necessary to realize this vision, particularly in the areas of decision support for software reuse and configuration control. Results for the Computer-aided Prototyping System project are summarized. (3/4/97)

Computational Molecular Nanotechnology at NAS, presented by Al Globus, Jie Han, Creon Levit, and Deepak Srivastava, of the computational nanotechnology group at Ames Research Center. The status and rationale of the NAS Systems Division/Ames effort, with progress to date and future plans, are presented. The project's future focus will address basic scientific issues of fullerene technology, such as appropriate force field potentials and fullerene tribology. These advances will require the design and simulation of additional computer and mechanical components. (2/27/97)

Modeling Transport in Nano Devices, presented by M. P. Anantram, of UCLA's School of Electrical Engineering. Technology improvements have provided a great impetus to device miniaturization and research labs that can build nanodevices with feature sizes in the tens-of-Angstrom range. At these length scales, quantum mechanics plays an increasingly important role in understanding the underlying device

physics. An overview of the trend in miniaturization and recent advances in quantum device physics is presented. (2/24/97)

Performance of a Computational Fluid Dynamics Code on NEC and Cray Supercomputers: Beyond 10 Gigaflops, presented by Ferhat Hatay of Ames Research Center. The impact of different computer architectures and high-performance computing approaches is given for an important engineering and scientific problem.

The model solves the viscous compressible flow equations for high-speed wall-shear layer flows, and the code has performed well on vector machines as well as massively parallel platforms. Performance characteristics were obtained for various supercomputers from NEC Corp. and Cray Research Inc. Comparative studies are presented for sustained performance achieved during the actual production runs for single and multiprocessing environments.

A scalable performance of 1.2 gigaflops was achieved on each processor of the NEC SX-4 and 550 megaflops on each processor of a CRAY C90. (2/11/97)

More Technical Reports Published Online

Here are summaries of new NAS Technical Reports published in 1997.

The Automated Instrumentation and Monitoring System (AIMS) Version 3.2 Users' Guide--*Jerry Yan, Melisa Schmidt, and Cathy Schulbach*

Careful analysis of execution traces can help computer designers and software architects uncover system behavior and take advantage of specific application characteristics and hardware features. This report describes AIMS 3.2, a software toolkit that facilitates performance evaluation of parallel applications on multiprocessors. Aside from use as a prototype for developing techniques for instrumenting, monitoring, and visualizing parallel program execution, AIMS is also being incorporated into the run-time environments of various hardware testbeds to evaluate their impact on user productivity. (NAS-97-001)

A Portable MPI Implementation of the SPAI Preconditioner in ISIS++--*Stephen T. Barnard and Robert L. Clay*

A parallel Message Passing Interface implementation of the Sparse Approximate Inverse (SPAI) preconditioner is described. A highly effective preconditioner, SPAI is inherently parallel because it computes columns/rows of the preconditioning matrix independently. However, several problems must be addressed for an efficient MPI implementation: load balance, latency hiding, and one-sided communication. The effectiveness, efficiency, and scaling behavior of this implementation is demonstrated for different platforms. (NAS-97-002)

Analysis and Optimization of Software Pipeline Performance on MIMD Parallel Computers--*Rob F. Van der Wijngaart, Sekhar R. Sarukkai, and Pankaj Mehra*

Observations show that fine-grained software and pipelines on MIMD parallel computers with asynchronous communication experience dynamic load imbalances that cause delays, in addition to the expected pipeline fill time. An analytical model that explains these load imbalances is presented. The results of applying this optimization to general pipeline algorithms on the Intel iPSC/860, Intel Paragon, and IBM SP2, as well as to pipelined tri-diagonal equation solvers on the Intel Paragon and the IBM SP2, are presented. (NAS-97-003)

RANS-MP: A Portable Parallel Navier-Stokes Solver--*Rob F. Van der Wijngaart and Maurice Yarrow*

RANS-MP, a new parallel implementation of a single-grid Navier-Stokes solver using the diagonalized

Beam-Warming approximate-factorization scheme, is presented. This first release of the completely rewritten solver, based on OVERFLOW, employs several optimizations. Results of realistic wing computations on the IBM SP2 are presented. In addition to high performance, an important feature of RANS-MP is its true portability -- thanks to the use of portable message-passing and I/O libraries, MPI and MPI-I/O. (NAS-97-004)

Technical Reports published at the end of last year are also available, including:

Molecular Dynamics Simulation of Carbon Nanotube Based Gears--*Jie Han, Al Globus, Richard Jaffe, and Glenn Deardorff*

Molecular dynamics was used to investigate the properties and design space of molecular gears fashioned from carbon nanotubes, with teeth added via a benzene reaction known to occur in Csub_60. A modified, parallelized version of Brenner's potential was used to model inter-atomic forces within each molecule. A Leonard-Jones 6-12 potential was used for forces between molecules. Videos and atomic trajectory files are presented in xyz format. (NAS-96-017)

NanoDesign: Concepts and Software for a Nanotechnology Based on Functionalized Fullerenes--*Al Globus and Richard Jaffe*

Diamondoid nanotechnology is not physically accessible through straightforward extensions of current laboratory techniques. Proposed is a form of nanotechnology based on functionalized fullerenes and carbon nanotube-based gears, with teeth added through a benzene reaction known to occur with Csub_60. The gears are single-walled carbon nanotubes with appended o-benzene groups for teeth. This paper describes the molecular design techniques, rationale, and implementing software for fullerenes. (NAS-96-019)

May - June 1997
Vol. 2, No. 24

Executive Editor: Marisa Chancellor

Editor: Jill Dunbar

Senior Writer: Elisabeth Wechsler

Contributing Writers: David H. Bailey, Jie Han, David Kao, George B. Myers, Parkson Wong

Image Coordinator: Chris Gong

Other Contributors: Richard Anderson, Johnny Chang, Bob Ciotti, James Donald, Gail Felchle, Michael Gerarld-Yamasaki, Mary Hultquist, Nateri Madavan, Bill Nitzberg, Marcia Redmond, Cathy Schulbach, Leigh Ann Tanner, Dave Tweten, Jose Zero

Editorial Board: Cristy Brickell, Marisa Chancellor, Jill Dunbar, Harry Waddell, Nateri Madavan, George Myers, Fay Pattee, Patrick Moran, Fred Templin, Elisabeth Wechsler

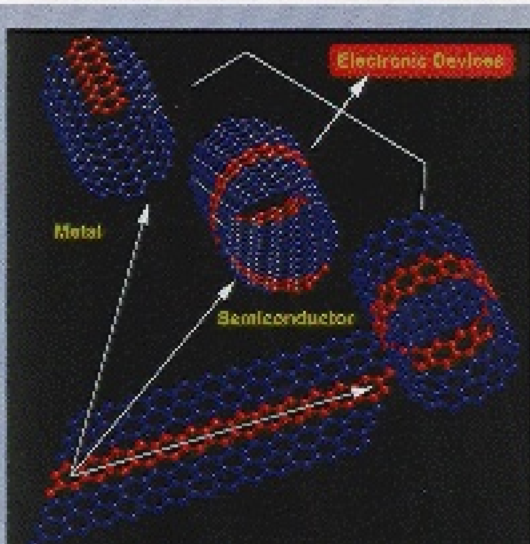
Thanks to Editorial Board members who contributed their ideas, time, and effort to NAS News during the last year: Nick Cardo, Chris Gong, David Kao, Mary Hultquist, and Chuck Niggley.

NAS NEWS
www.nas.nasa.gov/NAS/NASnews
Volume 2, Number 04 May – June 1997

Researchers at the NAS Facility are examining the role of the fullerene structure—a form of carbon—in creating electronic devices for the next century. In this article, scientists Jie Han and Subhash Saini give a bird's-eye view of the transition from conventional CMOS-based semiconductor devices to quantum devices, and discuss how fullerenes may benefit 'nanoelectronics' within the next ten years.

Fullerene-based Nanoelectronic Devices Show 'Tremendous Potential'

by Jie Han and Subhash Saini



Need for Leading-Edge Systems Spurs New Parallel Testbed

by Elisabeth Wechsler

THIS ISSUE